Manejo de Historias Clínicas.

Juan Alarcón, Johan Patiño, Adrian Benavides, Natalia Guevara, Daniel Rodriguez, Juan Saldaña, Jorge Alejandro Torres.

No. de equipo de trabajo: 1.

I. INTRODUCCIÓN

En el presente informe se plantea el enfoque desde el cual se abordará el problema seleccionado y las estructuras de datos (temporalmente lineales) que se implementarán para llevar a cabo en el desarrollo de este.

Se analizarán las funcionalidades las cuales debe cumplir nuestro programa y en base a eso se seleccionarán las estructuras más óptimas para el desarrollo del objetivo establecido.

Por último, se mostrarán los primeros bocetos de interfaces de usuario y un primer prototipo de la funcionalidad Autenticación de Usuarios, además de algunas pruebas de funcionamiento en las que se ejemplifica la correcta ejecución del programa.

II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

Errores en la programación, la desconexión en distintos ámbitos de la medicina, el traslado de pacientes entre EPS, bases de datos gigantescas para sistemas pobremente preparados para la magnitud de estos, la conectividad y la accesibilidad al internet de la universidad, entre otros tantos factores, hacen que la necesidad de organizar y acceder eficientemente a una gran cantidad de datos médicos sea un desafío fundamental en el ámbito de la salud dada la necesidad del equilibrio entre la cantidad de datos a trabajar y la eficiencia necesaria a la hora del acceso a estos.

Así pues, para la resolución del mismo, se propone desarrollar un programa especializado el cual utilizara diferentes estructuras de datos como lo son los stacks, colas, listas enlazadas, arreglos y árboles de búsqueda binaria para poder abordar esta problemática, donde se busca la priorización del tiempo de búsqueda para la cantidad de datos a trabajar.

Estas estructuras de datos nos permitirán administrar de manera efectiva la información de los pacientes en tratamiento al mismo tiempo asegurando la seguridad e integridad de cada uno de estos datos. La implementación de las estructuras de datos stack se utilizarán para registro cronológico de eventos médicos mientras que la cola nos ayudará a gestionar los turnos y citas de los pacientes. Por último los arreglos y árboles nos permitirán realizar búsquedas y análisis mientras que las listas enlazadas nos darán una mayor flexibilidad en la organización de datos complejos.

III. USUARIOS DEL PRODUCTO DE SOFTWARE

El programa reconoce a tres tipos de usuarios: administradores, médicos y pacientes.

Los administradores son los encargados de registrar nuevos usuarios o eliminarlos del sistema. Pueden ingresar pacientes, doctores e incluso otros administradores.

Los doctores pueden acceder a la información de cualquier paciente a voluntad y agregar, eliminar y modificar historias clínicas libremente. Además, tienen acceso a una cola de aquellos pacientes que tengan citas asignadas con ellos.

Por último, los pacientes tendrán la opción de consultar las diferentes versiones de sus historias clínicas, asimismo tendrán la elección de agendar citas médicas con cualquier doctor.

IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

1. Autenticación de Usuarios.

El propósito de esta funcionalidad es proporcionar acceso al sistema y privilegios acorde al rol de los usuarios.

Solo podrán acceder aquellos que los administradores ya hayan registrado. Para crear nuevos usuarios los administradores entran a su menú y definen usuario, contraseña, rol, tipo de documento, número de documento, nombre y fecha de nacimiento. Por supuesto, también pueden eliminarlos y modificar su información.

Esta información se guarda en un ArrayList y una lista enlazada para evitar la pérdida de datos cuando se deje de ejecutar el programa, constantemente se va guardando en una base de datos que se vuelve a cargar, se lee, y se convierte nuevamente en un arreglo.

Se debe llenar en un formato de inicio de sesión el usuario (número de cédula) y contraseña, datos que posteriormente se verifican en un ArrayList que además guarda el rol de la persona. Según este último el usuario pasará al menú correspondiente.

2. Encolamiento de Pacientes.

Esta funcionalidad se encarga de administrar de manera efectiva la atención de pacientes mediante la utilización de una estructura de cola (Queue) para garantizar un orden de atención eficiente. Los pacientes en su inicio de sesión, se irán a la cola de todos los médicos registrados en el programa.

Por otro lado, el personal médico puede acceder a la cola de pacientes encolados en una ventana separada, lo que les permite ver uno a uno a los pacientes registrados y acceder inmediatamente a sus historiales clínicos.

Es importante aclarar que en la ventana de registro, todos los demás datos serán guardados en un arreglo, los cuales podrán recuperarse para futuras historias clínicas, mientras que solo el nombre quedará dentro de la cola y será el dato que podrá visualizar el doctor.

3. Búsqueda de usuarios:

La ventana de administrador ofrece una funcionalidad esencial que permite buscar y gestionar usuarios en el sistema en función de su nombre o número de cédula. Este proceso es posible gracias a la capacidad del programa de leer y escanear un archivo de texto (formato .txt) que contiene todas las credenciales de los usuarios registrados. Al cargar estos datos desde el archivo y almacenarlos en un arreglo, se crea una fuente centralizada de información. Esto facilita la comparación de datos y la búsqueda de usuarios específicos en función de los criterios proporcionados, como el nombre o la cédula del usuario. Una vez que se localiza el usuario deseado, el administrador tiene la posibilidad de consultar, modificar o eliminar sus datos, lo que brinda un control completo sobre la gestión de usuarios en el sistema.

Funcionalidades a Futuro:

4. Búsqueda de antecedentes clínicos:

Dentro de un árbol binario de búsqueda se podrá organizar de tal manera que cada nodo del árbol podría representar un tipo de antecedente clínico o una categoría (por ejemplo, enfermedades cardiovasculares, procedimientos quirúrgicos, alergias, etc.), y los subnodos podrían contener información más específica. Cada paciente estaría asociado con este árbol de búsqueda binaria, y se podrían agregar antecedentes médicos en los nodos correspondientes según sea necesario.

5. Creación de historiales médicos:

Si los historiales médicos se componen principalmente de eventos cronológicos o cambios a lo largo del tiempo, podríamos utilizar una lista enlazada para representar cada evento. Esto facilita la inserción y eliminación de eventos en el historial.

V. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

Al comenzar, a los usuarios primero deben ingresar por el menú de Log In:

Figura [1] : Ventana de inicio de sesión para el usuario.

Ahora, según el rol del usuario se le redirigirá a su menú correspondiente.

Los administradores podrán acceder a las siguientes opciones:

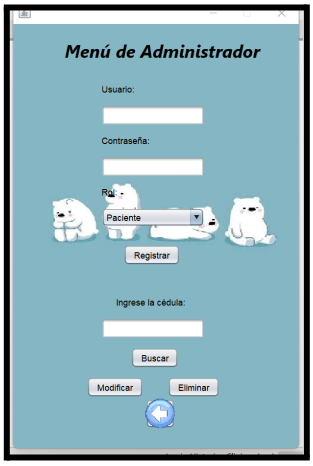

Figura [2] : Ventana de administrador para agregación de usuarios.

Esta sería el apartado de pacientes:

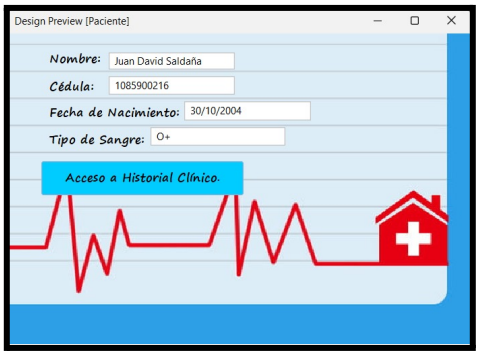

Figura [3] : Ventana de usuario sobre información personal.

Por último, el menú de los doctores:

Figura [4] : Ventana sobre lista de doctores registrados.

VI. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

El desarrollo de este software se llevó a cabo en NetBeans, en parte debido a la familiaridad de varios miembros del equipo con esta plataforma. Además, su interfaz de desarrollo gráfico es bastante intuitiva y sencilla de usar, por lo que el GUI fue relativamente fácil de crear.

Sin embargo, para futuras implementaciones y mejoras, estamos considerando la posibilidad de migrar hacia IntelliJ IDEA, un IDE que ofrece características avanzadas y una experiencia de desarrollo más potente y completa a la hora de la implementación y trabajo en las diferentes herramientas.

Por otro lado, el software utiliza un archivo de texto plano (archivo txt) para almacenar los datos de los pacientes, esto proporciona una solución externa la cual evita la pérdida de los datos obtenidos a la hora del cierre de la aplicación.. No obstante, estamos evaluando la posibilidad de migrar a un sistema de gestión de bases de datos en un futuro próximo. Esta transición permitiría un almacenamiento más robusto, seguro y eficiente de la información de los pacientes, mejorando así la escalabilidad y la gestión de datos a medida que el software continúa evolucionando y creciendo.

Está previsto utilizar una computadora de mesa con un sistema operativo Windows que proporcionará la plataforma necesaria para ejecutar la aplicación de manera eficiente, garantizando una experiencia de usuario fiable.

VII. PROTOTIPO DE SOFTWARE INICIAL

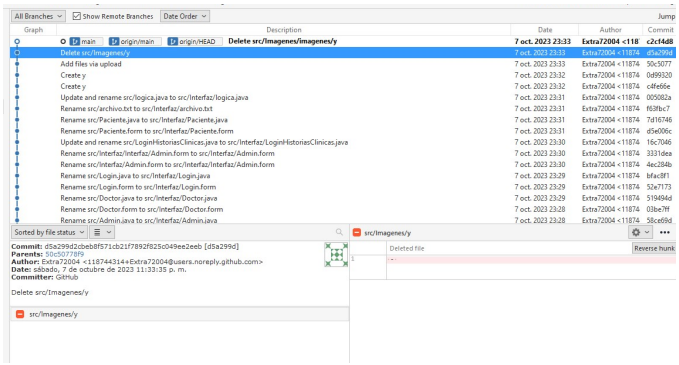

Figura [5] : Ventana de visualización del repositorio SourceTree.

Se puede acceder al repositorio con el siguiente link:

https://github.com/NataliaGuevaraR/Manejo_De_Historias_Clinicas.git

VIII. DISEÑO, IMPLEMENTACIÓN Y APLICACIÓN DE LAS ESTRUCTURAS DE DATOS

ArrayList:

- Creación: Al momento de ingresar al menú de inicio de sesión se busca el archivo txt que contiene las credenciales, se inicializa el arreglo y se guarda la información. En caso de estar vacío, de todas formas se deja inicializado y el administrador puede guardar nuevos usuarios directamente en el arreglo. Al cerrar el programa se sustituyen los datos que tiene el archivo txt por lo del arreglo actual.
- Inserción de un solo dato: Un administrador puede crear un usuario con los datos: usuario, contraseña y rol.
- Actualización de un solo dato: Un administrador puede actualizar los datos de un usuario.
- Eliminación de un solo dato: Un administrador puede eliminar los registros de usuarios.
- Búsqueda de un solo dato: El administrador puede consultar un paciente por su número de cédula o nombre.
- Consulta de todos los datos:
- Almacenamiento de los datos: Se almacenan en un archivo .txt.

LinkedList:

- Creación: Al momento de iniciar sesión como médico o paciente se buscará el archivo txt que contiene los datos de los pacientes. En caso de estar vacío, si se es paciente se cargan los datos de dicho paciente y en caso de ser doctor se deja inicializado. Al cerrar el

programa se sustituyen los datos que tiene el archivo txt por los de la lista actual.

- Inserción de un solo dato: Cuando un paciente registrado por primera vez inicia sesión con sus credenciales se carga su información a la lista.
- Actualización de un solo dato: Por un lado cada paciente puede editar su propia información personal mientras que los administradores la de cualquier paciente.
- Eliminación de un solo dato: Cuando un administrador elimina el registro de un paciente se eliminan sus datos.
- Búsqueda de un solo dato: El administrador puede consultar los datos de los pacientes verificados filtrados por número de documento.
- Consulta de todos los datos: El administrador y el médico pueden consultar todos los datos de los pacientes.
- Almacenamiento de los datos: Se almacenan en un archivo txt.

Stack:

- Creación: Se crea a partir de la LinkedList
- Consulta de todos los datos: Para mostrar a los médicos la lista de pacientes de forma descendente por orden de registro.

Queue:

- Creación: Se crea vacía.
- Inserción de un solo dato: Cuando un paciente agenda una nueva cita, se encola.
- Eliminación de un solo dato: El doctor puede desencolar a sus pacientes en su ventana, hasta que la cola quede vacía.
- Consulta de todos los datos: Muestra todos los pacientes pendientes en la cola de citas.

IX. INFORMACIÓN DE ACCESO AL VIDEO DEMOSTRATIVO DEL PROTOTIPO DE SOFTWARE

En el siguiente link se puede encontrar la versión preliminar en funcionamiento:

https://youtu.be/HoANWphoOGY

X. ROLES Y ACTIVIDADES

INTEGRANTE	ROL(ES)	ACTIVIDADES REALIZADAS (Listado)
Jorge Torres	Líder	Clasificación de los datos para su uso en las listas Stack y Queue
		Planeación inicial
Natalia Guevara	Coordinadora	Programación de reuniones presenciales y virtuales.
		Creación del repositorio.
	Observadora	Planeación de funcionamiento inicial.
		Redacción informe.
Adrian Benavidez	Investigador	
	Animador	
Juan Saldaña	Observador	Encolamiento de Pacientes.
		Redacción Informe .
	Animador	Interfaz Preliminar.
		Creación de Arreglo con Usuario.s
Camilo Patiño	Secretario	Redacción Informe
		Planeación de reuniones presenciales.
Daniel Rodríguez	Tecnico	Interfaces Gráficas de todos los tipos de usuarios.
		Sistema de Log In, Registro, y autenticación de Usuarios.
	Observador	Redacción de Informe.
Juan Alarcón	Experto	Revisión general uso de estructuras de datos

Tabla [1] : Roles de los Integrantes

XI. DIFICULTADES Y LECCIONES APRENDIDAS

Uno de los primeros retos fue el de organizar las ideas. Se llevaron a cabo varias reuniones en las que todos los miembros del grupo propusieron como implementar x o y funcionalidad, y con cuál estructura de datos se debería hacer.

Se propusieron algunas metodologías muy buenas. Pero la cosa es que eso no fue una lluvia de ideas, sino una tormenta, por lo que esas ideas se perdieron en la tormenta.

No fue hasta que se comenzó a hacer el informe y a programar algunos aspectos que esas ideas se aterrizaron y se empezaron a discutir las más factibles.

Para la próxima entrega hay que hacer las cosas con más calma, tiempo, y orden.

El siguiente reto se presentó en la parte de la permanencia de información. El login de usuarios requiere que los usuarios ya registrados queden guardados permanentemente, o al menos hasta que el administrador decida eliminarlos.

Al momento de cerrar el programa y volver a iniciarlo, toda la información que contiene el ArrayList empleado se pierde. Por lo que hubo que buscar un método para almacenar esa información permanentemente.

Se considera la opción de, a futuro, usar bases de datos para solventar este problema. De momento, se investigó otra forma

de hacerlo y se solucionó guardando toda la información de credenciales en un archivo txt.

Por último, uno de los mayores inconvenientes fue el trabajo colaborativo. Todos los miembros están acostumbrados a distintos entornos de desarrollo, así que exportar e importar código fue imposible en algunos casos.

Además, no todos están familiarizados con el uso de gitHub, por lo que eso también dificulta un poco las cosas. Hay que practicar más con la aplicación.

XII. X REFERENCIAS BIBLIOGRÁFICAS

[1] Weiss, M.A.: *Data Structures and Algorithm Analysis in C++, 4th Edition*, Pearson/Addison Wesley, 2014.

[2] Hernández, Z.J. y otros: *Fundamentos de Estructuras de Datos. Soluciones en Ada, Java y C++*, Thomson, 2005.

[3] Shaffer, Clifford A.: *Data Structures and Algorithm Analysis in C++, Third Edition*, Dover Publications, 2013. (En línea.)

[4] Campos Laclaustra, J.: *Apuntes de Estructuras de Datos y Algoritmos*, segunda edición, 2018. (En línea.)

[5] Martí Oliet, N., Ortega Mallén, Y., Verdejo López, J.A.: Estructuras de datos y métodos algorítmicos: 213 ejercicios resueltos. 2ª Edición, Ed. Garceta, 2013.

[6] Joyanes, L., Zahonero, I., Fernández, M. y Sánchez, L.: Estructura de datos. Libro de problemas, McGraw Hill, 1999.